Hierarchy of Asynchronous Automata

Rémi MORIN

L.R.I., Bât. 490, Université de Paris Sud, 91405 Orsay Cedex, France

Abstract

We study a natural notion of communication structure associated with asynchronous automata: we characterize which transition systems are isomorphic to an asynchronous automaton w.r.t. a given communication structure. For that, we present an algorithm to split global states into local states of communicating processes, similar to the regional technique for the synthesis problem of Petri nets. Our main result is an axiomatic criterion for the communication structures which decompose the same class of transition systems; this allows us to characterize and compare several particular classes of asynchronous automata. An immediate corollary of this study is a generic extension of Zielonka's theorem. We finally apply this method to asynchronous automata which describe systems of processes that communicate through shared memories.

Key words: Asynchronous (cellular) automaton, (asynchronous) transition system, Mazurkiewicz traces, shared memory, distributed system, concurrency.

Introduction

The study of distributed systems uses several kinds of models such as Petri nets, asynchronous transition systems, Mazurkiewicz traces or event structures. The comparison of these models essentially relies on the characterization of their intrinsic expressive power with the help of semantics between distinct levels of abstraction [2,16,5]; this allows to study such and such feature in the most convenient model. For instance, the so-called *synthesis problem* consists in deciding which transition systems correspond to the marking graph of particular classes of Petri nets; this issue was first tackled by Ehrenfeucht and Rozenberg who introduced the theory of regions [8] and since then it has been developed in many ways (see, e.g., [1,4]). In [12], we extended this issue to synchronized products of automata and deterministic classical asynchronous automata [17].

In this paper, we first address the question of characterizing several other classes of asynchronous automata studied in the literature, namely the 2-asynchronous automata [3], the cellular asynchronous automata [18] or the exclusive-read-owner-write asynchronous automata [6]. Each of these particular subclasses is characterized by some properties of the associated *communication structures* which specify read and write alphabets for each process. In Section 2, we show how to decide whether a transition system may be decomposed as a set of communicating processes w.r.t. a given communication structure. We obtain a criterion similar to the regional axioms known for Petri nets. We should recall here that such regional characterizations recently proved to be useful to synthesize distributed systems from abstract specifications [1,4]. Similarly, it may be the case that the regional criterion obtained here could help to build protocols for systems of communicating processes satisfying some given specifications.

Next, we associate to each particular model a *single* specific communication structure. Therefore, in order to compare the expressive power of particular classes of asynchronous automata, we simply compare their associated communication structures. For this, we introduce a *simulation pre-order* over the set of communication structures. Our main result asserts that two communication structures decompose the same class of transition systems if and only if they simulate one another (Th. 3.4).

As an immediate corollary, we observe that none of the generalizations of classical asynchronous automata really extends their expressive power in terms of underlying transition systems; moreover this study may be related to Zielonka's theorem [17,18] and lead to the following generalization: any recognizable trace language is the language of a finite asynchronous automaton w.r.t. to (almost) any communication structure.

Finally, in Section 4, we illustrate the generality of our approach and present a specific communication structure for the asynchronous automata which describe systems of processes communicating with shared memories [10]. This completes the results of [12] which studies the transition systems that correspond to distributed systems composed of processes and communication channels.

1 Basic Notions

Zielonka's asynchronous automata are a useful model for concurrent systems; they give theoretically a finite distributed implementation of any recognizable trace language [17] and provide a framework for describing the behavior of distributed systems or parallel machines [10]. The main aspect of this model lies in the representation of the system as a set of interacting sequential components; therefore, global states appear as sets of local states: this is the fundamental difference from the so-called asynchronous systems [2], trace automata [16] or automata with concurrency relations [7] which are more abstract representations of concurrent systems.

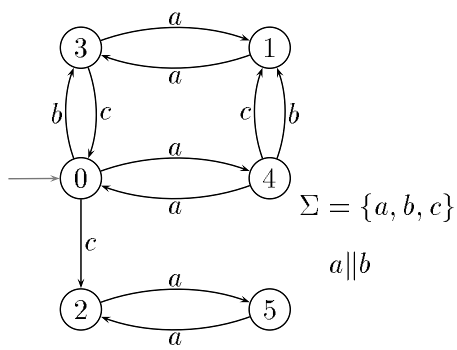


Fig. 1. Independent automaton $\mathcal{A}_1$

1.a Asynchronous Automata

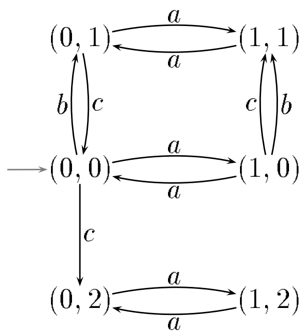
Throughout the paper, we will consider a fixed alphabet Σ equipped with a symmetric and irreflexive relation $\|$ over Σ; this *independence relation* tells which actions can occur simultaneously — or in any order — at distinct localities of the network. Now, a very common way to represent the behaviors of a concurrent system is simply to specify an automaton which describes its possible sequential executions: the concurrent runs of the system correspond then to the associated Mazurkiewicz' traces [5] w.r.t. the independence relation.

Definition 1.1 *An* independent automaton *over* $(\Sigma, \|)$ *is a structure* $\mathcal{A} = (Q, S, \Sigma, \longrightarrow, F, \|)$ *where* Q *is a set of states,* $S \subseteq Q$ *is the non-empty set of initial states,* $F \subseteq Q$ *is the set of final states, and* $\longrightarrow \subseteq Q \times \Sigma \times Q$ *is a set of labeled transitions. As usual, we will write* $q \xrightarrow{a} q'$ *instead of* $(q, a, q') \in \longrightarrow$.

In this paper, we assume that each state q is *reachable*, i.e. there is an integer $n \in \mathbb{N}$, some states q_0,..., q_n and actions a_1,..., a_n such that $q_0 \xrightarrow{a_1} q_1$,..., $q_{n-1} \xrightarrow{a_n} q_n = q$ and $q_0 \in S$. As usual we say that two independent automata over $(\Sigma, \|)$ are *isomorphic* if there is a one-to-one correspondence between their sets of states which preserves the initial states, the final states, and the labeled transitions. Note that, opposite to [2,16], we do not assume here any link between the underlying transition system of an independent automaton and its associated independence relation. Also, the independent automata studied in this paper may be non-deterministic.

Example 1.2 *Figure 1 describes an independent automaton* $\mathcal{A}_1$ *with six states, represented by circles, and eleven transitions, represented by labeled arrows. There is no final state and only one initial state decorated with a grey arrow.*

As explained above, a less abstract useful model for concurrent systems is provided by asynchronous automata. We adopt here the definition introduced by Diekert and Métivier [6] which extends both original asynchronous automata [17] and cellular asynchronous automata [18]. Essentially, an asynchronous automaton consists of a set K of processes which synchronize on particular actions according to their respective read and write alphabets.



$\Sigma = \{a, b, c\}$

$a \| b$

t	$R(t)$	$W(t)$
a	$\{x\}$	$\{x\}$
b	$\{y\}$	$\{y\}$
c	$\{x, y\}$	$\{y\}$

Fig. 2. Asynchronous automaton $\mathcal{A}_2$

Definition 1.3 *An* asynchronous automaton *of rank K over $(\Sigma, \|)$ is an independent automaton $\mathcal{A} = (Q,\ S,\ \Sigma,\ \longrightarrow,\ F,\ \|)$ for which there are*

- *a family of sets of local states $(Q_k)_{k \in K}$ such that $Q \subseteq \prod_{k \in K} Q_k$,*
- *a* read relation $R \subseteq \Sigma \times K$ *and and a* write relation $W \subseteq \Sigma \times K$ *such that*
  S_0*: for each action $a \in \Sigma$, $W(a) \subseteq R(a)$,*
- *for each action $a \in \Sigma$, a* transition function $\delta_a \subseteq \prod_{k \in R(a)} Q_k \times \prod_{k \in W(a)} Q_k$,

such that the transitions of $\mathcal{A}$ respect the following synchronization rule

$$(q_k)_{k \in K} \xrightarrow{a} (q'_k)_{k \in K} \Leftrightarrow \begin{cases} \forall k \notin W(a), q'_k = q_k \\ \left((q_k)_{k \in R(a)}, (q'_k)_{k \in W(a)}\right) \in \delta_a \end{cases}$$

and the independence relation satisfies the two following conditions:

S_1*: $\forall a, b \in \Sigma$: $a \| b \Rightarrow W(a) \cap R(b) = \emptyset$;*
S_2*: $\forall a, b \in \Sigma$: $R(a) \cap R(b) = \emptyset \Rightarrow a \| b$.*

Here, the behavior of the system consists of transitions which *synchronize* some particular processes: in order to perform a transition a, the local states of the processes in its *read domain* $R(a)$ are read and according to their values and the transition function δ_a some local changes of states are enabled for the processes of its *write domain* $W(a)$. This procedure is atomic. Now, condition S_1 indicates that two actions a and b can occur independently — hence in any order — only if action a cannot change the local states read by b. Therefore, any asynchronous automaton satisfies the usual forward and independent diamond properties [15,2,13,12].

FD: $q_1 \xrightarrow{a} q_2 \wedge q_1 \xrightarrow{b} q_3 \wedge a \| b \Rightarrow \exists q_4 \in Q, q_2 \xrightarrow{b} q_4 \wedge q_3 \xrightarrow{a} q_4$.

ID: $q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_4 \wedge a \| b \Rightarrow \exists q_3 \in Q, q_1 \xrightarrow{b} q_3 \xrightarrow{a} q_4$.

The second requirement S_2 for the independence relation insures that whenever two actions involve distinct components, they can occur independently; this corresponds to the intuition of concurrency in systems of communicating processes.

Example 1.4 *Figure 2 describes an asynchronous automaton $\mathcal{A}_2$ with two processes x and y. Process x has two states; it can switch from* 0 *to* 1 *and back by performing a transition a. Process y has three states; it can change its local state from* 0 *to* 1 *with a transition b; the read domain $R(c)$ of c contains both x and y: depending on their local states, process y can perform a transition c.*

We remark here that $\mathcal{A}_1$ and $\mathcal{A}_2$ of Figures 1 and 2 are *isomorphic*: there is a bijection between their states which preserves and reflects labeled transitions and initial or final states. Unfortunately, most independent automata are not isomorphic to an asynchronous automaton; that is why the construction of a finite asynchronous automaton associated with a given recognizable trace language is often not easy [17,11,18,9].

1.b Particular Models

First, in the original model, there is no difference between the read domain or the write domain of any action; therefore, we will say that an asynchronous automaton is *classical* if $R = W$. In [3], it is proved that one can restrict to classical asynchronous automata for which each process can perform only two actions without loss of expressive power; the *2-asynchronous* automata of rank K are such that $\forall k \in K$: $\mathrm{Card}(R^{-1}(k)) \leq 2$. Next, in [18], Zielonka introduced cellular asynchronous automata: here, each process is associated with one specific action which is repeatedly performed according to the local states of its neighbors; the network is formalized by the independence relation: $a \not\parallel b$ means that process a executes an action a depending on the local state of process b. Formally, an asynchronous automaton of rank K is *cellular* if $K = \Sigma$ and for each action a: $W(a) = \{a\}$ and $R(a) = \{b \in \Sigma \mid b \not\parallel a\}$. Finally, in [6], the more general model used in this paper is introduced (Def. 1.3) together with the following properties.

Definition 1.5 *Let $\mathcal{A}$ be an asynchronous automaton; $\mathcal{A}$ satisfies the* Exclusive-Read *property if $\forall a, b \in \Sigma$, $a \| b \Leftrightarrow R(a) \cap R(b) = \emptyset$. It satisfies the* Concurrent-Read *property if $\forall a, b \in \Sigma$, $a \| b \Leftrightarrow R(a) \cap W(b) = R(b) \cap W(a) = \emptyset$. Finally, $\mathcal{A}$ is* Owner-Write *if $W(a) \cap W(b) \neq \emptyset \Rightarrow a = b$.*

The Concurrent-Read property corresponds to the so-called Bernstein conditions whereas the Exclusive-Read property forbids simultaneous readings of the local state of one process by two other ones. Clearly, any classical asynchronous automaton satisfies the Exclusive-Read and the Concurrent-Read properties; furthermore, any cellular asynchronous automaton is Owner-Write and Concurrent-Read. Note also that the asynchronous automaton of Fig. 2 is not classical because $R(c) \neq W(c)$. We should stress finally that Zielonka introduced in [5, chap 7] some generalized asynchronous automata which are not required to satisfy Axiom S_0 of Def. 1.3. However, they are easily shown to be isomorphic to Concurrent-Read asynchronous automata.

2 Synthesis of Asynchronous Automata

In this section, we relate the model of independent automata with the model of asynchronous automata. Precisely we characterize which independent automata can be decomposed into sequential communicating processes; this study extends to non-deterministic and more general asynchronous automata the technique sketched in [12]. We show moreover why the underlying algorithm is much simpler when we restrict ourself to deterministic independent automata.

2.a Realizable Communication Structures

The distributed structure and the communication between the processes of an asynchronous automaton of rank K is based on its read relation R and its write relation W; this constitutes the *communication structure* of each asynchronous automaton (Def. 1.3).

Definition 2.1 *A* communication structure *of rank K is a pair $\tau = (R, W)$ of relations over $\Sigma \times K$ satisfying axioms S_0, S_1 and S_2 of Def. 1.3.*

Note that if action a belongs to the *write alphabet* $W^{-1}(k)$ and action b belongs to the corresponding *read alphabet* $R^{-1}(k)$ then, by S_1, a and b are dependent; hence, each write alphabet is a clique[1] of the dependence graph $(\Sigma, \not\parallel)$. Moreover, S_2 insures that each action appears in at least one read alphabet.

Thus, in order to decompose an independent automaton $\mathcal{A}$ one has to choose a communication structure τ such that $\mathcal{A}$ is isomorphic to an asynchronous automaton associated with τ. Such a communication structure is called *realizable* for $\mathcal{A}$.

Definition 2.2 *A communication structure τ is* realizable *for an independent automaton $\mathcal{A}$ if $\mathcal{A}$ is isomorphic to an asynchronous automaton whose communication structure is τ.*

Now, given a communication structure τ, we aim to know which independent automata are isomorphic to an asynchronous automaton whose communication structure is τ. Clearly, these independent automata should fulfill the classical diamond properties of so-called asynchronous systems [2,12]. Yet, a more precise criterion is given by the following result.

Lemma 2.3 *Let $\mathcal{A}$ be an independent automaton and $\tau = (R, W)$ be a communication structure of rank K. Then, τ is realizable for $\mathcal{A}$ iff there are equivalences $(\equiv_k)_{k \in K}$ over the states of $\mathcal{A}$ such that*

- NS_1: $q \xrightarrow{a} q' \wedge k \notin W(a) \Rightarrow q \equiv_k q'$;
- NS_2: $\forall q_1, q_2 \in Q$: $(\forall k \in K, q_1 \equiv_k q_2) \Rightarrow q_1 = q_2$;

---

[1] A clique of $(\Sigma, \not\parallel)$ is a subset Δ of Σ such that $\forall a, b \in \Delta, a \not\parallel b$.

NS_3: $\forall a \in \Sigma$, $\forall q_1, q_2 \in Q$: $q_1 \xrightarrow{a} q_1' \wedge (\forall k \in R(a), q_1 \equiv_k q_2) \Rightarrow \exists q_2' : q_2 \xrightarrow{a} q_2' \wedge \forall k \in W(a), q_1' \equiv_k q_2'$.

Proof. We call *co-orientation* any family of equivalences $(\equiv_k)_{k\in K}$ satisfying axioms NS_1, NS_2, and NS_3.

(i) Assume that $\mathcal{A}$ is an asynchronous automaton of rank K. Then the family of equivalences $\left(\equiv_k^{\mathcal{A}}\right)_{k\in K}$ defined by

$$(q_k)_{k\in K} \equiv_j^{\mathcal{A}} (q_k')_{k\in K} \Leftrightarrow q_j = q_j'$$

over the communication structure $\tau^{\mathcal{A}}$ of $\mathcal{A}$ is a co-orientation. Clearly $\left(\equiv_k^{\mathcal{A}}\right)_{k\in K}$ is a co-orientation over τ. Moreover, if $\mathcal{A}_1$ and $\mathcal{A}_2$ are isomorphic and if $\mathcal{A}_1$ admits a co-orientation over τ then $\mathcal{A}_2$ admits a co-orientation over τ too.

(ii) Let τ be a communication structure. If τ admits a co-orientation of $\mathcal{A}$ then $\mathcal{A}$ is isomorphic to an asynchronous automaton whose communication structure is τ.

Let $\mathcal{A} = (Q, S, \Sigma, \longrightarrow, F, \|)$ be an independent automaton and $(\equiv_k)_{k\in K}$ be a co-orientation of $\mathcal{A}$ over τ. For each $k \in K$, we consider Q_k the set of equivalence classes of states w.r.t. $\equiv_k$. For each $a \in \Sigma$, we write $R(a) = R^\tau(a)$, $W(a) = W^\tau(a)$ and consider $\delta_a \subseteq \prod_{k\in R(a)} Q_k \times \prod_{k\in W(a)} Q_k$ such that

$$\left((q_k)_{k\in R(a)}, (q_k')_{k\in W(a)}\right) \in \delta_a \Leftrightarrow \exists q \xrightarrow{a} q' \text{ in } \mathcal{A}, \begin{cases} \forall k \in R(a) : \lfloor q \rfloor_k = q_k \\ \forall k \in W(a) : \lfloor q' \rfloor_k = q_k' \end{cases}$$

where $\lfloor q \rfloor_k$ denotes the equivalence class of q w.r.t. $\equiv_k$. Finally, we note

$$S^\dagger = \{(\lfloor q \rfloor_k)_{k\in K} \mid q \in S\}$$

and

$$F^\dagger = \{(\lfloor q \rfloor_k)_{k\in K} \mid q \in F\}.$$

We consider then the asynchronous automaton

$$\mathcal{A}^\dagger = (\prod_{k\in K} Q_k, S^\dagger, \Sigma, \longrightarrow^\dagger, F^\dagger, \|)$$

whose transitions are given by

$$(q_k)_{k\in K} \xrightarrow{a}{}^\dagger (q_k')_{k\in K} \Leftrightarrow \begin{cases} \forall k \notin W(a), q_k' = q_k \\ \left((q_k)_{k\in R(a)}, (q_k')_{k\in W(a)}\right) \in \delta_a \end{cases}$$

We show that the map

$$\begin{aligned} \varphi : \mathcal{A} &\rightarrow \mathcal{A}^\dagger \\ q &\mapsto (\lfloor q \rfloor_k)_{k\in K} \end{aligned}$$

is a isomorphism. It is clear that φ preserves the initial and final states. Moreover, if $q \xrightarrow{a} q'$ then, by NS_1, $\varphi(q) \xrightarrow{a}{}^\dagger \varphi(q')$. Hence φ is a

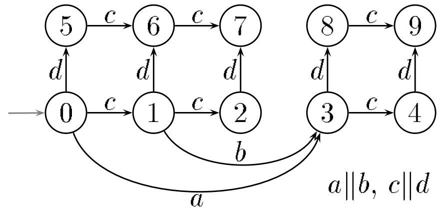


Fig. 3. Independent automaton $\mathcal{A}_3$

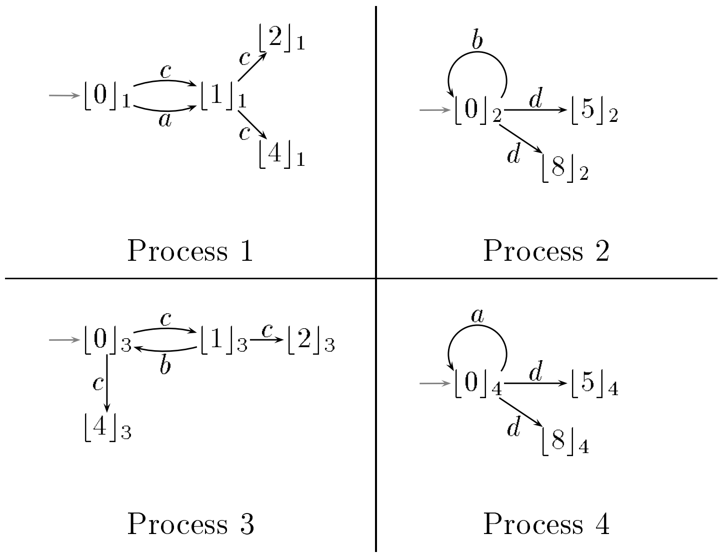


Fig. 4.

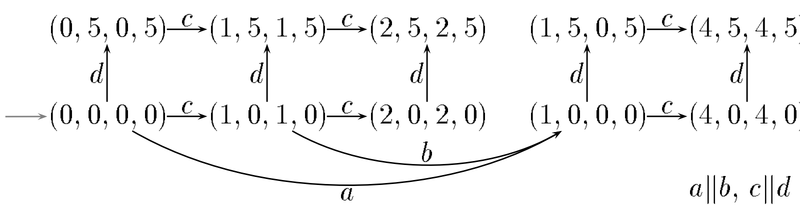


Fig. 5. Asynchronous automaton $\mathcal{A}_4$

morphism. By NS_2, φ is one-to-one. On the other hand, if $\varphi(q_1) = q_1^\dagger \xrightarrow{a}^\dagger (q'_k)_{k\in K}$ in $\mathcal{A}^\dagger$ then there is a transition $q_3 \xrightarrow{a} q_4$ in $\mathcal{A}$ such that for each $k \in R(a)$, $q_1 \equiv_k q_3$ and $\forall k \in W(a)$: $\lfloor q_4 \rfloor_k = q'_k$. By NS_3, there is $q_2 \in Q$ such that $q_1 \xrightarrow{a} q_2$ and $\forall k \in W(a)$: $q_2 \equiv_k q_4$, i.e., $\lfloor q_2 \rfloor_k = q'_k$; now, for each $k \notin W(a)$, $q'_k = \lfloor q_1 \rfloor_k = \lfloor q_2 \rfloor_k$, hence $\varphi(q_2) = (q'_k)_{k\in K}$. So, φ is a bijection between the states of $\mathcal{A}$ and those of $\mathcal{A}^\dagger$; it is even an isomorphism.

Example 2.4 *We consider here the independent automaton $\mathcal{A}_3$ described in Fig. 3. We wonder if $\mathcal{A}_3$ is isomorphic to an asynchronous automaton* *whose communication structure of rank* $[1,4]$ *is* $\tau = (R, W)$ *where* $R^{-1}(1) = W^{-1}(1) = \{a, c\}$, $R^{-1}(2) = W^{-1}(2) = \{b, d\}$, $R^{-1}(3) = W^{-1}(3) = \{b, c\}$, *and* $R^{-1}(4) = W^{-1}(4) = \{a, d\}$. *For each process* $k \in [1,4]$, *we use an equivalence* $\equiv_k$ *which identifies global states that correspond to the same local state. Therefore, if* $q \xrightarrow{a} q'$ *and* $k \notin W(a)$ *then* $q \equiv_k q'$. *In other words, Axiom* NS_1 *leads to the four processes of Fig. 4 where* $\lfloor q \rfloor_k$ *denotes the equivalence class of state* q *w.r.t* $\equiv_k$ *and the transitions of each process are reduced to its write alphabet. Precisely,*

- *for Process* 1: $0 \equiv_1 5$, $1 \equiv_1 6$, $2 \equiv_1 7$, $3 \equiv_1 8$, $4 \equiv_1 9$ *and* $1 \equiv_1 3$;
- *for Process* 2: $0 \equiv_2 1 \equiv_2 2$, $7 \equiv_2 6 \equiv_2 5$, $8 \equiv_2 9$ *and* $0 \equiv_2 3 \equiv_2 4$;
- *for Process* 3: $0 \equiv_3 5$, $1 \equiv_3 6$, $2 \equiv_3 7$, $3 \equiv_3 8$, $4 \equiv_3 9$ *and* $0 \equiv_3 3$;
- *for Process* 4: $0 \equiv_4 1 \equiv_4 2$, $7 \equiv_4 6 \equiv_4 5$, $8 \equiv_4 9$ *and* $1 \equiv_4 3 \equiv_4 4$.

We observe now that the independent automaton $\mathcal{A}_3$ *is deterministic (Fig. 3); therefore, in order to apply the technique detailed in the proof above and get a deterministic asynchronous automaton we should identify also the states 5 and 8 in the processes 2 and 4. The point is that in the initial state, both processes 2 and 4 can execute two transitions* d. *In that way, we obtain the asynchronous automaton* $\mathcal{A}_4$ *of Fig. 5 which is isomorphic to the independent automaton* $\mathcal{A}_3$ *of Fig. 3.*

2.b Deterministic Independent Automata

Thus in order to split the states of an independent automaton into local states of communicating processes, one has simply to choose an adequate family of equivalences of (global) states and check that Axioms NS_1, NS_2 and NS_3 are fulfilled. The naive underlying algorithm is clearly non-deterministic; however we show now that this result leads to a deterministic polynomial algorithm (in the size of the automaton) when we restrict ourself to *deterministic* independent automata, that is to say such that there is only one initial state and $q \xrightarrow{a} q' \wedge q \xrightarrow{a} q'' \Rightarrow q' = q''$. In the end of this section, we consider a fixed communication structure $\tau = (R, W)$ of rank K and we assume that $\mathcal{A}$ is a *deterministic* independent automaton. Under these assumptions, the simpler criterion will rely on a *least* family of equivalences according to the following definition.

Definition 2.5 *An* orientation *of* $\mathcal{A}$ *over* τ *is a family of equivalences* $(\equiv_k)_{k \in K}$ *over the states of* $\mathcal{A}$ *such that*

DE_1: $q \xrightarrow{a} q' \wedge k \notin W(a) \Rightarrow q \equiv_k q'$;

DE_2: $$\left.\begin{array}{c} q_1 \xrightarrow{a} q_1' \wedge q_2 \xrightarrow{a} q_2' \\ \forall k \in R(a), q_1 \equiv_k q_2 \end{array}\right\} \Rightarrow \forall k \in W(a), q_1' \equiv_k q_2'.$$

We say that an orientation $(\equiv_k)_{k\in K}$ *is smaller than an orientation* $\left(\equiv_k^\dagger\right)_{k\in K}$ *if for each* $k \in K$, $\equiv_k \subseteq \equiv_k^\dagger$.

We remark now that this partial order of orientations admits a least element.

Lemma 2.6 *There is a least orientation of* $\mathcal{A}$ *over* τ.

Proof. First, τ admits some orientations, namely the full orientation such that for each $k \in K$, $\equiv_k = Q \times Q$. We consider for each $k_0 \in K$ the equivalence $\equiv_{k_0}^\dagger$ defined by: $q_1 \equiv_{k_0}^\dagger q_2$ if, and only if, for each orientation $(\equiv_k)_{k\in K}$ of τ, $q_1 \equiv_{k_0} q_2$. It is clear that if $q \xrightarrow{a} q'$ and $k_0 \notin W(a)$ then $q \equiv_{k_0}^\dagger q'$; in fact, for all orientation $(\equiv_k)_{k\in K}$ of τ, $\forall k \notin W(a)$: $q \equiv_k q'$; in particular, $q \equiv_{k_0} q'$. Hence $q \equiv_{k_0}^\dagger q'$. Assume now that $q_1 \xrightarrow{a} q_1'$ and $q_2 \xrightarrow{a} q_2'$ in $\mathcal{A}$ and that for each $k \in R(a), q_1 \equiv_k^\dagger q_2$. Let $k_0 \in K$ be such that $k_0 \in W(a)$. For all orientation $(\equiv_k)_{k\in K}$ of τ, for each $k \in R(a), q_1 \equiv_k q_2$ hence $\forall k \in W(a)$: $q_1' \equiv_k q_2'$. Consequently $q_1' \equiv_{k_0}^\dagger q_2'$. Therefore $\left(\equiv_k^\dagger\right)_{k\in K}$ is an orientation of τ.

One can easily compute this particular orientation when $\mathcal{A}$ and τ are finite. Starting with the trivial equivalences $(\equiv_k)_{k\in K}$ for which $q \equiv_k q'$ iff $q = q'$ we first apply DE_1 and next repeatedly apply DE_2 until this second requirement is fulfilled. This minimal orientation is now used for our first main result.

Theorem 2.7 *Let* $(\equiv_k)_{k\in K}$ *be the least orientation of* $\mathcal{A}$ *over* τ. *The communication structure* τ *is realizable for* $\mathcal{A}$ *iff the two following conditions are satisfied:*

DS_1*:* $\forall q_1, q_2 \in Q$: $[\forall k \in K, q_1 \equiv_k q_2] \Rightarrow q_1 = q_2$;

DS_2*:* $\forall a \in \Sigma$, $\forall q_1, q_2 \in Q$: $\left[q_1 \xrightarrow{a} \wedge \forall k \in R(a), q_1 \equiv_k q_2\right] \Rightarrow q_2 \xrightarrow{a}$.

Proof. If $(\equiv_k)_{k\in K}$ satisfies DS_1 and DS_2 then clearly it satisfies NS_1, NS_2 and NS_3 so τ is realizable for $\mathcal{A}$. Conversely, we assume now that $\mathcal{A}$ is realizable for $\mathcal{A}$. Then there are equivalences $\left(\equiv_k^\dagger\right)_{k\in K}$ satisfying NS_1, NS_2 and NS_3. Because $\mathcal{A}$ is deterministic, these equivalences form an orientation of $\mathcal{A}$ which satisfies DS_1 and DS_2. Now, because $(\equiv_k)_{k\in K}$ is smaller than $\left(\equiv_k^\dagger\right)_{k\in K}$ it satisfies DS_1 and DS_2 too. To see this, consider first q_1 and q_2 such that $\forall k \in K$: $q_1 \equiv_k q_2$. Then, $\forall k \in K$: $q_1 \equiv_k^\dagger q_2$ hence $q_1 = q_2$. Consider now two states q_1 and q_2 and an action a such that $q_1 \xrightarrow{a}$ and $\forall k \in R(a)$: $q_1 \equiv_k q_2$; then $\forall k \in R(a)$: $q_1 \equiv_k^\dagger q_2$ hence $q_2 \xrightarrow{a}$.

We remark here the similarity between conditions DS_1 and DS_2 and the so-called *regional separation axioms* used for the synthesis problem of Petri nets [8,1].

Example 2.8 *We consider here again the independent automaton* $\mathcal{A}_1$ *of Fig. 1 and the communication structure of rank* $K = \{x, y\}$ *for which* $W(a) = R(a) = \{x\}$, $W(b) = R(b) = \{y\}$, $W(c) = \{y\}$ *and* $R(c) = \{x, y\}$. *Ap-*

plying conditions DE_1 *and* DE_2 *leads to the following identifications: by* DE_1, $0 \equiv_x 2 \equiv_x 3$ *and* $1 \equiv_x 4$*; by* DE_2, $4 \equiv_x 5$ *because* $R(a) = \{x\}$ *and* $0 \equiv_x 2$*. On the other hand,* $0 \equiv_y 4$, $1 \equiv_y 3$ *and* $2 \equiv_y 5$*. We easily check that these equivalences satisfy* DS_1 *and* DS_2*; therefore,* $\mathcal{A}_1$ *is isomorphic to an asynchronous automaton.*

We remark finally that the problem of deciding whether a deterministic independent automaton is isomorphic to an asynchronous automaton is NP: if $\mathcal{A}$ has n states and m actions, we need to find a realizable communication structure with $n^2.(m+1)$ processes in order to fulfill DS_1 and DS_2. More precisely, for each pair of distinct states (q_1, q_2), there must be a process k such that $\neg(q_1 \equiv_k q_2)$ in order to fulfill DS_1. Furthermore, for each pair of states (q_1, q_2) and each action a such that $q_1 \xrightarrow{a}$ and $\neg(q_2 \xrightarrow{a})$, there must be a process $k \in R(a)$ such that $\neg(q_1 \equiv_k q_2)$ in order to fulfill DS_2. Thus we need less than $n^2 + n^2.m$ processes.

3 Classification of Asynchronous Automata

We now come to the core of the paper. As detailed in Section 1, several variations of the original notion of asynchronous automaton have been introduced in the literature; for instance, 2-asynchronous automata [3], cellular asynchronous automata [18], or exclusive-read owner-write asynchronous automata [6] keep the expressive power of classical asynchronous automata: they correspond to all recognizable trace languages. Yet, these models are not *structurally equivalent*, as remarked previously by Pighizzini [14]. In this section, we show how the study of realizable communication structures of Section 2 can be applied to compare these classes of asynchronous automata and leads to a simple criterion for structurally equivalent models. First, one can naively associate to each particular model a corresponding communication structure.

Example 3.1 *We consider first the communication structure* $\tau^{cel} = (R^{cel}, W^{cel})$ *of rank* $K = \Sigma$ *over* $(\Sigma, \|)$ *such that for each action* $a \in \Sigma$, $W^{cel}(a) = \{a\}$ *and* $R^{cel}(a) = \{b \in \Sigma \mid a \not\| b\}$*. Clearly, an independent automaton* $\mathcal{A}$ *is isomorphic to a cellular asynchronous automaton iff* τ^{cel} *is realizable for* $\mathcal{A}$ *(Theorem 2.7). Thus* τ^{cel} *characterizes the class of cellular asynchronous automata.*

Next, different models will be compared w.r.t. the relation between their associated communication structure. In that way, we will establish for instance that any owner-write asynchronous automaton is isomorphic to a *cellular* asynchronous automaton and that any asynchronous automaton is isomorphic to a *classical* asynchronous automaton. The results of this section hold for possibly non-finite state and non-deterministic asynchronous automata.

3.a Equivalent Communication Structures

We naturally associate to any communication structure the class of independent automata for which it is realizable. This leads to the following natural notion of *equivalent* communication structures.

Definition 3.2 *Two communication structures τ^1 and τ^2 are* equivalent *if for each independent automaton $\mathcal{A}$, τ^1 is realizable for $\mathcal{A}$ iff τ^2 is realizable for $\mathcal{A}$.*

In other words, two communications structures are equivalent if they correspond to the same class of independent automata. We give here a simple axiomatic criterion for equivalent communication structures; this essentially relies on the following *simulation pre-order.*

Definition 3.3 *A communication structure $\tau^1 = (R^1, W^1)$ of rank K* simulates *a communication structure $\tau^2 = (R^2, W^2)$ of rank J if*

$$\forall j \in J,\ \forall a \in \Sigma\text{: } j \in R^2(a) \Rightarrow \exists k \in R^1(a),\ (W^2)^{-1}(j) \subseteq (W^1)^{-1}(k).$$

This abstract notion of *simulation* is justified by our main result below (Th. 3.4): τ^1 simulates τ^2 iff any asynchronous automaton with communication structure τ^2 is isomorphic to another asynchronous automaton with communication structure τ^1.

Theorem 3.4 *Let τ^1 and τ^2 be two communication structures; the following conditions are equivalent:*

(i) *τ^1 simulates τ^2;*

(ii) *for each independent automaton $\mathcal{A}$: if τ^2 is realizable for $\mathcal{A}$ then τ^1 is also realizable for $\mathcal{A}$;*

(iii) *for each independent automaton $\mathcal{A}$: if $\mathcal{A}$ is isomorphic to an asynchronous automaton whose communication structure is τ^2 then $\mathcal{A}$ is also isomorphic to an asynchronous automaton whose communication structure is τ^1.*

Proof. By Definition 2.2, (ii) ⇔ (iii). We note $\tau^1 = (R^1, W^1)$ and $\tau^2 = (R^2, W^2)$ two communication structures of rank K and J respectively. First, (i) ⇒ (ii): by Lemma 2.3, we can consider some equivalences $\left(\equiv_j^2\right)_{j\in J}$ satisfying NS_1, NS_2, and NS_3. We define the family of equivalences $(\equiv_k^1)_{k\in K}$ by

$$q \equiv_k^1 q' \Leftrightarrow \left[\forall j \in J : (W^2)^{-1}(j) \subseteq (W^1)^{-1}(k) \Rightarrow q \equiv_j^2 q'\right]$$

and check easily that it satisfies NS_1 and NS_2.

- NS_1: Consider $q \xrightarrow{a} q'$ and $k \in K$ such that $k \notin W^1(a)$. For $j \in J$ such that $(W^2)^{-1}(j) \subseteq (W^1)^{-1}(k)$: we have $k \notin W^1(a)$ hence $j \notin W^2(a)$ and $q \equiv_j^2 q'$. Hence, $q \equiv_k^1 q'$.
- NS_2: Consider q_1 and q_2 such that $\forall k \in K$: $q_1 \equiv_k^1 q_2$. For each $j \in J$, there is a process $k \in K$ such that $(W^2)^{-1}(j) \subseteq (W^1)^{-1}(k)$; now $q_1 \equiv_k^1 q_2$ so $q_1 \equiv_j^2 q_2$. Hence $q_1 = q_2$.

We observe that it also satisfies NS_3: consider $a \in \Sigma$ and q_1, q_1', q_2 such that $q_1 \xrightarrow{a} q_1'$ and $\forall k \in R^1(a)$, $q_1 \equiv_k^1 q_2$. We consider first $j \in J$ such that $j \in R^2(a)$; there is $k \in K$ such that $\left(W^2\right)^{-1}(j) \subseteq \left(W^1\right)^{-1}(k)$ and $k \in R^1(a)$: hence $q_1 \equiv_k^1 q_2$ and $q_1 \equiv_j^2 q_2$. Therefore, there is q_2' such that $q_2 \xrightarrow{a} q_2'$ and $\forall j \in W^2(a)$, $q_1' \equiv_j^2 q_2'$. Consider $k \in K$ such that $k \in W^1(a)$; for each $j \in J$ such that $\left(W^2\right)^{-1}(j) \subseteq \left(W^1\right)^{-1}(k)$, if $j \in W^2(a)$ and then $q_1' \equiv_j^2 q_2'$; otherwise $j \notin W^2(a)$: $q_1' \equiv_j^2 q_1$ and $q_2' \equiv_j^2 q_2$; now $k \in W^1(a) \subseteq R^1(a)$ so $q_1 \equiv_k^1 q_2$, $q_1 \equiv_j^2 q_2$ and $q_1' \equiv_j^2 q_2'$. Thus, in any case, $q_1' \equiv_j^2 q_2'$. Hence $q_1' \equiv_k^1 q_2'$.Therefore τ^1 is also realizable for $\mathcal{A}$.

Now, $\neg$(i) $\Rightarrow \neg$(ii): there are $a_0 \in \Sigma$ and $j_0 \in R^2(a_0)$ such that $\forall k \in K$, $\left(W^2\right)^{-1}(j_0) \subseteq \left(W^1\right)^{-1}(k) \Rightarrow k \notin R^1(a_0)$. We build the *deterministic* automaton $\mathcal{A}$ with two states, 0 and 1, such that $0 \xrightarrow{a} 0$ if $a \neq a_0$ and $j_0 \notin W^2(a)$, $0 \xrightarrow{a} 1$ if $j_0 \in W^2(a)$, and for all $a \in \Sigma$, $1 \xrightarrow{a} 1$; we check that τ^2 is a realizable communication structure of $\mathcal{A}$ but not τ^1. *First, τ^2 is realizable for $\mathcal{A}$.* We consider the equivalences $\left(\equiv_j^2\right)_{j \in J}$ over the states 0 and 1 such that $j \neq j_0 \Leftrightarrow 0 \equiv_j^2 1$. We easily check that it satisfies DE_1, DE_2, DS_1, and DS_2. of $\mathcal{A}$. First, DE_1 is satisfied; otherwise there exists an action $a \in \Sigma$ such that $j_0 \notin W^2(a)$ and $0 \xrightarrow{a} 1$. Now, DE_2 is also satisfied; otherwise there are states q_1, q_1', q_2 and q_2', and an action $a \in \Sigma$ such that $q_1 \xrightarrow{a} q_1'$, $q_2 \xrightarrow{a} q_2'$, $\forall j \in R^2(a), q_1 \equiv_j^2 q_2$ but $j_0 \in W^2(a)$ and $q_1' \not\equiv_{j_0}^2 q_2'$; therefore $q_1' \neq q_2'$, $q_1 \neq q_2$ and $q_1 \equiv_{j_0}^2 q_2$ because $j_0 \in W^2(a) \subseteq R^2(a)$. Clearly, DS_1 is fulfilled because $\neg(0 \equiv_{j_0}^2 1)$. Finally, DS_2 is fulfilled too because each action is enabled in each state, except a_0 which is maybe not enabled in the initial state 0 if $j_0 \notin W^2(a_0)$; now if DS_2 is not satisfied then for all $j \in R^2(a_0)$, $0 \equiv_j^2 1$: in particular, $0 \equiv_{j_0}^2 1$. *Now, τ^1 is not realizable for $\mathcal{A}$.* Assume that τ^1 is realizable for $\mathcal{A}$; then there are equivalences $(\equiv_k^1)_{k \in K}$ over the states 0 and 1 which satisfy NS_1, NS_2, and NS_3. By NS_2, there is $k_0 \in K$ such that $\neg(0 \equiv_{k_0}^1 1)$ so, by NS_1, $\left(W^2\right)^{-1}(j_0) \subseteq \left(W^1\right)^{-1}(k_0)$, $k_0 \notin R^1(a_0)$ and $j_0 \notin W^2(a_0)$. For any $k \in K$, if $\neg(0 \equiv_k^1 1)$ then $\left(W^2\right)^{-1}(j_0) \subseteq \left(W^1\right)^{-1}(k)$ so $k \notin R^1(a_0)$; in other words $\forall k \in R^1(a_0)$: $0 \equiv_k^1 1$. Now $1 \xrightarrow{a_0} 1$ so $0 \xrightarrow{a_0} 1$ by NS_3. Yet, $j_0 \notin W^2(a_0)$.

We finally obtain the following useful criterion.

Corollary 3.5 *Two communication structures are equivalent (Def. 3.2) iff they simulate one another.*

Example 3.6 *Consider the communication structure $\tau^{ow} = (R^{ow}, W^{ow})$ of rank $K = \{(a,b) \mid a \not\| b\}$ such that $(a,b) \in W(c) \Leftrightarrow c = a$ and $(a,b) \in R(c) \Leftrightarrow c \in \{a,b\}$; then, by Theorem 3.4, an independent automaton $\mathcal{A}$ is isomorphic to a* Owner-Write *asynchronous automaton iff τ^{ow} is realizable for $\mathcal{A}$; furthermore τ^{ow} is equivalent to τ^{cel} of Example 3.1. Therefore any Owner-Write asynchronous automaton is isomorphic to a cellular asynchronous automaton. Conversely, Theorem 3.4 also enables us to prove easily a result of [6]: any cellular asynchronous automaton is isomorphic to an exclusive-read owner-write asynchronous automaton.*

Particular Model	Associated Communication Structure(s)
asynchronous automata (a.a.) classical a.a.	τ^{cla} over $\{\Delta \subseteq \Sigma \mid \Delta$ maximal clique of $(\Sigma, \not\parallel)\}$ such that $R^{cla}(a) = W^{cla}(a) = \{k \mid a \in k\}$
classical 2-asynchronous automata	τ^{cta} over $\{\{a,b\} \subseteq \Sigma \mid a \not\parallel b\}$ such that $R^{cta}(a) = W^{cta}(a) = \{k \mid a \in k\}$
Owner-Write a.a. cellular a.a.	τ^{cel}, τ^{ow} are equivalent (Examples 3.1 and 3.6)

Table 1
Some communication structures and their associated independent automata.

3.b Characterizations and Comparisons of Particular Models

Continuing the preceding example, many classes of asynchronous automata may be characterized by a specific communication structure with the help of Th. 3.4. For instance, we establish the characterizations of particular models detailed in Table 1. First, we obtain the result of [12]: an independent automaton $\mathcal{A}$ is isomorphic to a *classical* asynchronous automaton iff τ^{cla} is realizable w.r.t. $\mathcal{A}$. Next we claim that *any communication structure is simulated by* τ^{cla}; therefore, according to Th. 3.4, an independent automaton is isomorphic to an asynchronous automaton iff τ^{cla} is realizable w.r.t. $\mathcal{A}$. Moreover, we obtain the noteworthy following result: none of the extensions of classical asynchronous automata really extends their expressive power.

Corollary 3.7 *Any asynchronous automaton is isomorphic to a* classical *asynchronous automaton.*

Consequently any generalized asynchronous automaton of [5, chap. 7] is also isomorphic to a classical asynchronous automaton.

Theorem 3.4 enables us to prove easily that an independent automaton $\mathcal{A}$ is isomorphic to a classical 2-asynchronous automaton iff the communication structure τ^{cta} defined in Table 1 is realizable for $\mathcal{A}$. Now, we observe that τ^{cla} simulates τ^{cta} which simulates τ^{cel} but *none of the converses holds* — as soon as $(\Sigma, \not\parallel)$ admits a 3-clique. Therefore, we obtain the following *strict* inclusions of models (up to isomorphisms):

cellular a.a. $\subset$ classical 2-asynchronous automata $\subset$ (classical) a.a.

3.c Implementation of Recognizable Trace Languages

Due to Zielonka's theorem [18], any recognizable trace language is the language of a finite cellular asynchronous automaton. Now, according to the inclusion of models above, it is also the language of a finite classical 2-asynchronous automaton and of a finite classical automaton, as also previously established in [3,17]. In fact, Zielonka's theorem holds for many other communications structures.

Corollary 3.8 *Let L be a recognizable trace language over $(\Sigma, \|)$ and $\tau = (R, W)$ a communication structure such that $a \not\| b \Rightarrow W(a) \cap R(b) \neq \emptyset$. Then there exists a* finite *asynchronous automaton whose language is L and whose communication structure is τ.*

Proof. Clearly, τ simulates τ^{ow}; we simply use Zielonka's construction [18] and apply Theorem 3.4.

4 Asynchronous Shared Memory Systems

Asynchronous automata can describe many kinds of distributed systems or parallel machines; in this section, we focus on systems of processes which communicate through shared memories [10]. In this context, each synchronized action represents a particular process reading or writing the value of a specific memory; consequently, it involves *only two components* of the system: therefore, we will assume that for any synchronized action a, $\mathrm{Card}(R(a)) = 2$. The other actions are restricted to only one component; they can represent a local computation or an interaction with the environment: for technical convenience, we will assume here that the system admits at least one action d which is not a synchronization: $\mathrm{Card}(R(d)) = 1$. *We will also assume in this section that the dependence graph $(\Sigma, \not\|)$ is connected*; otherwise the system can be split into several parts which behave independently. Finally, the so-called "shared-memory" asynchronous automata introduced below satisfy the Exclusive-Read condition (Def. 1.5); this means that processes can read the value of a shared memory only one at a time.

Definition 4.1 *An asynchronous automaton $\mathcal{A}$ is said* shared-memory *if it satisfies the Exclusive-Read property and the two following conditions:*

- SM_1*: $\forall a \in \Sigma$: $\mathrm{Card}(R(a)) \leq 2$;*
- SM_2*: $\exists d \in \Sigma$, $\mathrm{Card}(R(d)) = 1$.*

For these asynchronous automata, the actions d which involve only one component of the system satisfy the following property: $\forall a, b \in \Sigma$, $a \not\| d \not\| b \Rightarrow a \not\| b$; such an action d will be called an *operation* of $(\Sigma, \|)$. Moreover, the family of read alphabets $(R^{-1}(k))_{k \in K}$ is a covering by cliques of the dependence graph $(\Sigma, \not\|)$ and each action appears in less than two read alphabets: we will say that $(R^{-1}(k))_{k \in K}$ is a *2-covering* of $(\Sigma, \|)$. That is why we will naturally focus in the end of this section on concurrent alphabets $(\Sigma, \|)$ which admit an operation and a 2-covering.

In order to characterize which independent automata correspond to a shared-memory asynchronous automaton, we will use the construction of the "optimal alphabets" introduced in [12].

Definition 4.2 *The set of* optimal alphabets *Ω is the least set of subsets of Σ such that*

O_1: *for all operation* d, $\{a \in \Sigma \mid a \not\| d\} \in \Omega$,

O_2: $\forall \Delta \in \Omega, \forall a, b \in \Sigma$, $(a \in \Delta \wedge b \notin \Delta \wedge a \not\| b) \Rightarrow \{c \not\| a \mid c \notin \Delta \vee c \heartsuit a\} \in \Omega$,

where $a \heartsuit b$ *means* $\forall c \in \Sigma : a \| c \Leftrightarrow b \| c$.

Note here that Ω is empty if there is no operation in $(\Sigma, \|)$.

Theorem 4.3 *Let* $(\Sigma, \|)$ *admit an operation and a 2-covering and* τ^{opt} *be the communication structure of rank* Ω *such that* $\forall a \in \Sigma$, $R(a) = W(a) = \{\Delta \in \Omega \mid a \in \Delta\}$. *An independent automaton* $\mathcal{A}$ *is isomorphic to a shared-memory asynchronous automaton iff* τ^{opt} *is a realizable communication structure of* $\mathcal{A}$.

Opposite to classical or cellular asynchronous automata, finite shared-memory asynchronous automata do not correspond to all recognizable trace languages [17,18,6] because their associated concurrent alphabet $(\Sigma, \|)$ have some particular properties: as explained above, they admit an operation and a 2-covering. Yet, we have the following converse.

Corollary 4.4 *Let* $(\Sigma, \|)$ *admit an operation and a 2-covering; any recognizable trace language* L *over* $(\Sigma, \|)$ *is obtained by a finite shared-memory asynchronous automaton.*

Proof. Let $\mathcal{A}$ be a finite cellular asynchronous automaton which recognizes L; then τ^{ow} is a realizable communication structure of $\mathcal{A}$ and τ^{opt} simulates τ^{ow}.

Conclusion

In this paper, we have introduced a correspondence between a natural notion of communication structure and some particular classes of asynchronous automata studied in the literature. On one hand, we characterized the structural properties of these classes up to isomorphisms, which allow to decide which transition systems can be split as systems of cooperating processes w.r.t. a given communication structure. On the other hand, we presented a simple axiomatic criterion for the communication structures associated to the same class of transition systems. We showed how this study leads to a generalization of Zielonka's theorem; however, it is still unclear to us whether it also holds for some other communication structures which do not satisfy the restriction that $a \not\| b \Rightarrow W(a) \cap R(b) \neq \emptyset$. Clearly, such an extension would not rely directly on Zielonka's construction.

Finally we applied this study to a subclass of asynchronous automata which corresponds to the widely used model of asynchronous shared memory [10]. A particular communication structure based on the optimal alphabets of [12] characterizes this model; moreover, an adaptation of Zielonka's theorem is also established. In this direction, and similarly to the work of Darondeau [4], an interesting problem would be to use the regional technique developed here to produce automatically protocols satisfying some given safety and liveness properties.

Acknowlegments.

Thanks to the constructive criticism of some anonymous referees on a previous version of this paper, we could improve the presentation of our results.

References

[1] BADOUEL E., BERNARDINELLO L., DARONDEAU PH. (1995) : *Polynomial algorithms for the synthesis of bounded nets*, CAAP, LNCS 915 647-679

[2] BEDNARCZYK M.A. (1987) : *Categories of Asynchronous Systems*, thesis, University of Sussex

[3] CORI R., SOPENA E., LATTEUX M., ROOS Y. (1988) : *2-asynchronous automata*, TCS 61, p. 93-102

[4] DARONDEAU PH. (1998) : *Deriving nets from languages*, Concur'98, LNCS 1466, p. 533-548

[5] DIEKERT V., ROZENBERG G. (1995) : *The Book of Traces*, World Scientific Publishing Co.

[6] DIEKERT V., MÉTIVIER Y. (1996) : *Partial Commutations and Traces*, LITP 96/04 Institut Blaise Pascal, Paris

[7] DROSTE, M. (1990) : *Concurrency, automata and domains*, ICALP'90, LNCS 443

[8] EHRENFEUCHT A., ROZENBERG G. (1990) : *Partial (Set) 2-structures*, Part II: State spaces of concurrent systems, Acta Informatica, Vol. **27**, p. 343-368

[9] KLARLUND N., MUKUND M., SOHONI M. (1994) : *Determinizing Asynchronous Automata*, LNCS **820**, p. 130-141

[10] LYNCH, N.A. (1996) : *Distributed Algorithms*, Morgan Kaufmann Publishers, Inc., San Francisco, California

[11] MÉTIVIER Y. (1987) : *An algorithm for computing asynchronous automata in the case of acyclic non-commutation graph*, ICALP'87, LNCS 267, p. 226-236

[12] MORIN R. (1998) : *Decompositions of Asynchronous Systems*, Concur'98, LNCS 1466, p. 549-564

[13] MUKUND M., NIELSEN M. (1992) : *CCS, Locations and Asynchronous Transition Systems*, Report DAIMI, Vol. **3**, N. 4, p. 443-478 and LNCS **652**, p. 328-341

[14] PIGHIZZINI G. (1994) : *Asynchronous automata versus asynchronous cellular automata*, TCS **132**, p. 179-207

[15] SHIELDS M.W. (1985) : *Concurrent Machines*, the computer journal, vol 28(5), p.449-465

[16] Stark E. W. (1989) : *Connections between a Concrete and an Abstract Model of Concurrent Systems*, LNCS 442, p.53-79

[17] Zielonka W. (1987) : *Notes on finite asynchronous automata*, Theoretical Informatics and Applications, vol. **21**, p. 99-135

[18] Zielonka W. (1989) : *Safe executions of recognizable trace languages by asynchronous automata*, LNCS 363, p. 278-289